AMENDMENT

Listing of Claims

Please amend the claims as follows:

1. (Previously Presented) A method for dynamic compiling, comprising:

loading byte-code on a digital information appliance, said byte-code suitable for including a tagged section;

identifying the tagged section of the byte-code; and

compiling the tagged section of byte-code;

wherein the tagged section is compiled when the byte-code is loaded so as to enable the digital information appliance to utilize the tagged section of byte-code without additional compiling of the tagged section of byte-code by the digital information appliance,

wherein the byte-code includes at least one dynamic base object, the at least one dynamic base object comprising an interface dynamic base object and an implementation dynamic base object that communicate with each other over a message bus.

2. (Original) The method as described in claim 1, further comprising:

encoding application source code to byte-code, the byte-code including code in a processor-independent form which is suitable for further analysis; and

tagging a section of the byte-code.

3. (Original) The method as described in claim 2, wherein the byte-code including code in the processor-independent form is suitable for further analysis including at least one of suitable for compiler optimization, suitable for processing by interpreters, and suitable for use in generation of binary instruction for the digital information appliance processing system.

4. (Original) The method as described in claim 2, wherein the section of the byte-code is tagged for being performance sensitive.

5. (Original) The method as described in claim 1, further comprising storing the compiled tagged section of byte-code in persistent storage.

6. (Original) The method as described in claim 1, wherein loading includes validating that the byte-code conforms with byte-code suitable for utilization by the digital information appliance.

7. (Previously Presented) A digital information appliance suitable for dynamic coupling, comprising:

a processor for implementing a program of instructions; and

a memory for storing the program of instructions, the program of instructions suitable for configuring the digital information appliance to load byte-code, said byte-code suitable for including a tagged section;

identify the tagged section of the byte-code; and

compile the tagged section of byte-code;

wherein the tagged section is compiled when the byte-code is loaded so as to enable the digital information appliance to utilize the tagged section of byte-code without additional compiling of the tagged section of byte-code by the digital information appliance,

wherein the byte-code includes at least one dynamic base object, the at least one dynamic base object comprising an interface dynamic base object and an implementation dynamic base object that communicate with each other over a message bus.

8. (Original) The digital information appliance as described in claim 7, further comprising:

encoding application source code to byte-code, the byte-code including code in a processor-independent form which is suitable for further analysis; and

tagging a section of the byte-code.

9. (Original) The digital information appliance as described in claim 8, wherein the byte-code including code in the processor-independent form is suitable for further analysis including at least one of suitable for compiler optimization,

suitable for processing by interpreters, and suitable for use in generation of binary instruction for a digital information appliance's processing system.

10. (Original) The digital information appliance as described in claim 7, wherein the section of the byte-code is tagged for being performance sensitive.

11. (Original) The digital information appliance as described in claim 7, further comprising storing the compiled tagged section of byte-code in persistent storage.

12. (Original) The digital information appliance as described in claim 7, wherein loading includes validating that the byte-code conforms with byte-code suitable for utilization by the digital information appliance.

13. (Currently Amended) A system for providing an execution environment that is suitable for dynamic compiling, comprising:

a memory device suitable for storing computer readable information;

a loader coupled to the memory device, the loader suitable for loading byte-code to the memory, said byte-code suitable for including a tagged section, the loader being capable of interpreting, just in time compiling, and pre-compiling;

an identifier coupled to the loader, the identifier suitable for identifying the tagged section of the byte-code;

a compiler coupled to the identifier;

wherein the identified tagged section is compiled by the compiler when the byte-code is loaded so as to enable the tagged section of byte-code to be utilized without additional compiling of the tagged section of byte-code;

<u>wherein the byte-code includes at least one dynamic base object, the at least one dynamic base object comprising an interface dynamic base object and an implementation dynamic base object that communicate with each other over a message bus.</u>

14. (Original) The system as described in claim 13, further comprising:

an encoder for encoding application source code to byte-code, the byte-code including code in a processor-independent form which is suitable for further analysis; and

a tagger for tagging a section of the byte-code.

15. (Original) The system as described in claim 14, wherein the byte-code including code in the processor-independent form is suitable for further analysis including at least one of suitable for compiler optimization, suitable for processing by an interpreter, and suitable for use in generation of binary instruction for a processing system.

16. (Original) The system as described in claim 14, wherein the section of the byte-code is tagged for being performance sensitive.

17. (Original) The system as described in claim 13, wherein the loader includes a validation utility for validating the byte-code conforms with byte-code suitable for utilization by the system.

18. (Previously Presented) A method for providing an execution environment in an information appliance network, comprising:

a) encoding an application source code in a processor independent byte-code;

b) tagging at least some portion of said processor independent byte-code; and

c) compiling at least some portion of said tagged processor independent byte-code,

wherein the independent byte-code includes at least one dynamic base object, the at least one dynamic base object comprising an interface dynamic base object and an implementation dynamic base object that communicate with each other over a message bus.

19. (Original) The execution environment for an information appliance network of claim 18, further including validating at least some portion of said processor independent byte-code.

20. (Original) The execution environment for an information appliance network of claim 18, wherein compiling includes identifying the portion of said tagged processor independent byte-code.

21. (Previously Presented) The method of Claim 1, wherein the message bus provides interprocessor communications within a system.

22. (Previously Presented) The method of Claim 1, wherein the message bus provides communications over the network.

23. (Previously Presented) The method of Claim 1, wherein the byte-code is processor independent.

24. (Previously Presented) The method of Claim 1, wherein the dynamic base object is programmed through a scripting language with run-time object invocation.

25. (Previously Presented) The method of Claim 1, wherein the interface dynamic base object and the implementation dynamic base object communicate bi-directionally.

26. (Previously Presented) The method of Claim 7, wherein the message bus provides interprocessor communications within a system.

27. (Previously Presented) The method of Claim 7, wherein the message bus provides communications over the network.

28. (Previously Presented) The method of Claim 7, wherein the byte-code is processor independent.

29. (Previously Presented) The method of Claim 7, wherein the dynamic base object is programmed through a scripting language with run-time object invocation.

30. (Previously Presented) The method of Claim 7, wherein the interface dynamic base object and the implementation dynamic base object communicate bi-directionally.

31. (Cancelled)

32. (Currently Amended) The system of Claim ~~31~~ 13, wherein the message bus provides interprocessor communications within a system.

33. (Currently Amended) The system of Claim ~~31~~ 13, wherein the message bus provides communications over the network.

34. (Currently Amended) The system of Claim ~~31~~ 13, wherein the interface dynamic base object and the implementation dynamic base object communicate bi-directionally.

35. (Previously Presented) The system of Claim 13, wherein the byte-code is processor independent.

36. (Currently Amended) The system of Claim ~~31~~ 13, wherein the dynamic base object is programmed through a scripting language with run-time object invocation.

37. (Previously Presented) The method of Claim 18, wherein the message bus provides interprocessor communications within a system.

38. (Previously Presented) The method of Claim 18, wherein the message bus provides communications over the network.

39. (Previously Presented) The method of Claim 18, wherein the independent byte-code is processor independent.

40. (Previously Presented) The method of Claim 18, wherein the dynamic base object is programmed through a scripting language with run-time object invocation.

41. (Previously Presented) The method of Claim 18, wherein the interface dynamic base object and the implementation dynamic base object communicate bi-directionally.

42. (Previously Presented) The method of Claim 1, wherein the interface dynamic base object includes multiple lines of code.

43. (Previously Presented) The method of Claim 1, wherein the digital information appliance is a thin network appliance.

44. (Previously Presented) The method of Claim 1, wherein the at least one dynamic base object is fully thread safe.